Banco de Dados
Indexação

Professor: Carlos Alberto
Email: carloswgama@gmail.com

Preparando Ambiente

- Para essa aula, vamos criar um novo usuário para montarmos um novo banco:

```
CREATE DATABASE
aula_index;
```

- Após criado, execute os comandos script disponibilizado na aula.

Indexação

Como é feito a Indexação dos dados?

- As informações contidas num banco de dados são gravadas/escritas em uma estrutura no disco, de modo a permitir que os dados se mantenham salvos e que possam ser lidos posteriormente.

- Para buscar um determinado valor salvo no disco, o SGBD poderá realizar uma busca sequencial, binária ou direta.

  - Busca Sequencial – É realizado uma busca em sequencia (normalmente na ordem em que foi inserido) até encontrar o conteúdo desejado. Ou seja um problema de complexidade O(n)

  - Busca Binária – Tenta quebrar a busca em problemas menores, resultando uma busca O(logn)

  - Busca Direta – O Sistema sabe aonde está a informada desejada e vai direto na linha informada. Com sua complexidade O(1)

Exemplo de acesso

- Criamos uma tabela usuários e inserimos 3 valores, obtendo o seguinte resultado:

```
CREATE TABLE usuarios (
   id  SERIAL PRIMARY KEY,
   nome TEXT NOT NULL,
   cpf CHAR(14)
);
```

id	nome	cpf
4	Carlos	111.111.111-11
5	João	222.222.222-22
6	Maria	333.333.333-33

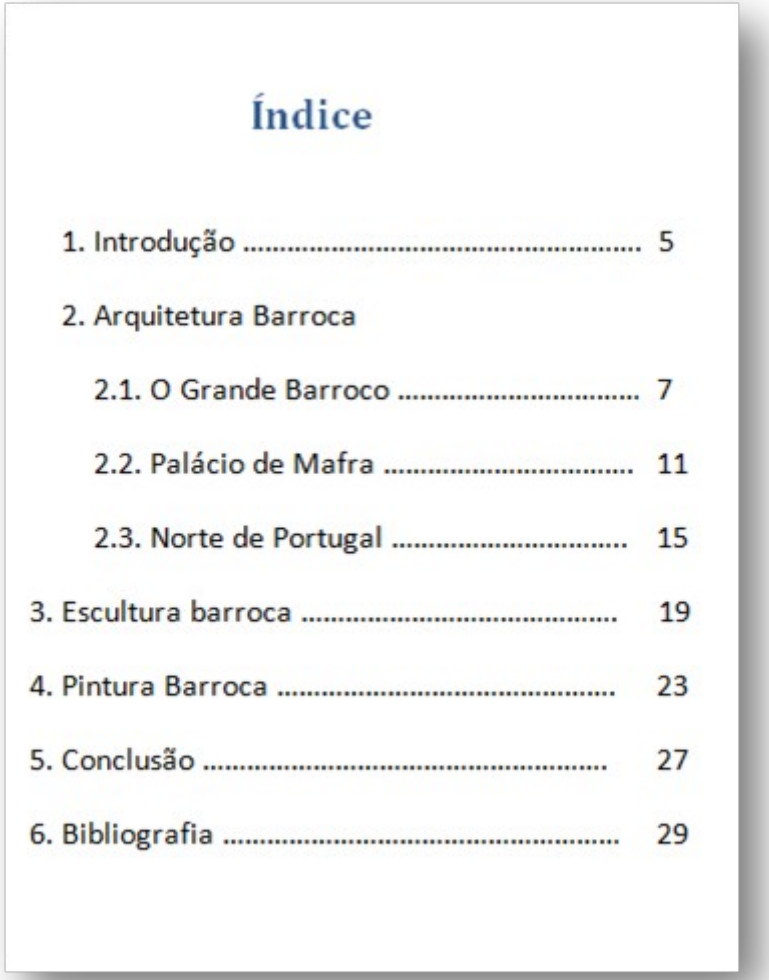
Exemplo de acesso sequencial

- Para realizar a busca pelo CPF ou nome não há nenhum índice associado a ele, desta forma será preciso realizar uma busca completa em toda tabela usuários até encontrar o CPF ou nome correspondente.

id	nome	cpf
4	Carlos	111.111.111-11
5	João	222.222.222-22
6	Maria	333.333.333-33

Indíce

- Mas com um campo que possui um índice, o processo é mais simples.
- Mas o que é um índice?
  - É um trabalho textual, é uma estrutura que aponta a página que um está localizado item (sumário por exemplo)
- Com banco de dados não é diferente.


Índice

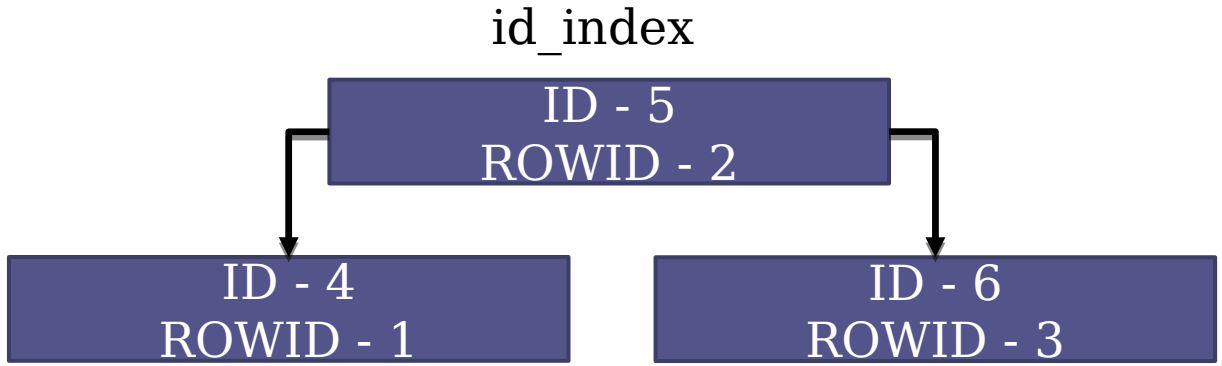
1. Introdução 5
2. Arquitetura Barroca
   - 2.1. O Grande Barroco 7
   - 2.2. Palácio de Mafra 11
   - 2.3. Norte de Portugal 15
3. Escultura barroca 19
4. Pintura Barroca 23
5. Conclusão 27
6. Bibliografia 29

- Um índice em BD armazena um valor em uma estrutura de dados não linear que possuía a ROWID (Localização da linha) de onde o valor é encontrado.

Exemplo de acesso direto

- Caso seja necessário realizar uma busca pelo ID, a busca ocorrerá de forma binária, uma vez que o ID é uma chave primária e sua posição já foi indexada em outro registro, tornando a busca mais rápida.

- Após achar o valor numa busca binária, ele irá informar a linha onde está o valor na estrutura principal, realizando uma busca direta.



id	nome	cpf
4	Carlos	111.111.111-11
5	João	222.222.222-22
6	Maria	333.333.333-33

É possível criar mais índices?

- Sim, em todo Banco de Dados Relacional, podemos criar outros índices além da chave primária.
- Esses índices vão proporcionar um aumento na velocidade da busca, para consultas muito lentas.
- Para isso devemos analisar qual campo da busca é utilizado com bastante frequência e criar um índice nele.

Criando índice de busca

- O comando para criar um índice vinculado a uma coluna de uma tabela é:

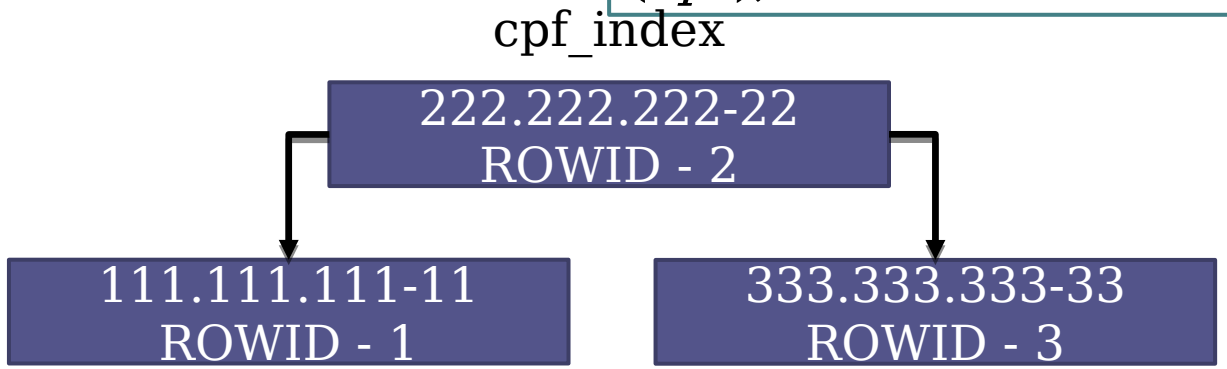
```
CREATE INDEX nome_index ON tabela (colu
na);
```

```
-- BUSCA OTIMIZADA
SELECT * FROM tabela WHERE coluna = "valor"
;
```

Criando índice de busca

- Como o CPF é um valor único e importante em buscas, é possível criar um indice para o CPF de modo a guardá-lo em outro registro, tornando o seu acesso de sequencial para direto.

```
CREATE INDEX cpf_index ON usuarios (cpf);
```



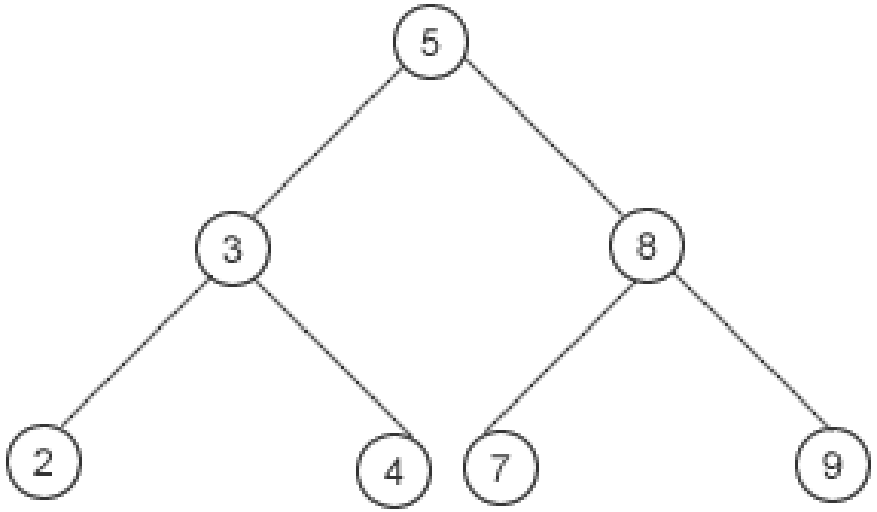
id	nome	cpf
4	Carlos	111.111.111-11
5	João	222.222.222-22
6	Maria	333.333.333-33

Tipos de Índices

- O PostgreSQL é capaz de criar índices em diversos formatos, usando os seguintes algoritmos de buscas diferentes:
  - B-tree
  - Hash
  - GiST
  - GIN
  - Entre outros

Tipos de Índices – B-Tree

- B-tree – É o padrão quando o tipo de índice não é informado. Armazena os dados num formato de árvore balanceada. Por tanto, pode trabalhar bem com buscas (selects) analisando índices maiores ou menores.



Tipos de Índices – B-Tree

- Ao criar um campo indexado com o algoritmo B-tree, sempre que fizermos uma busca usando as operações abaixo, a busca será otimizada:
  - <
  - <=
  - =
  - >=
  - >

- Exemplo:
  - SELECT jogador, pontuacao FROM jogadores WHERE pontuacao<= 100;
  - SELECT jogador, pontuacao FROM jogadores WHERE pontuacao = 100;

Tipos de Índices – B-Tree

- B-tree – Para a criação de um index de um campo de uma tabela com o algoritmo B-Tree usamos:

```
CREATE INDEX nome_index ON tabela (campo_da_tabela);
```

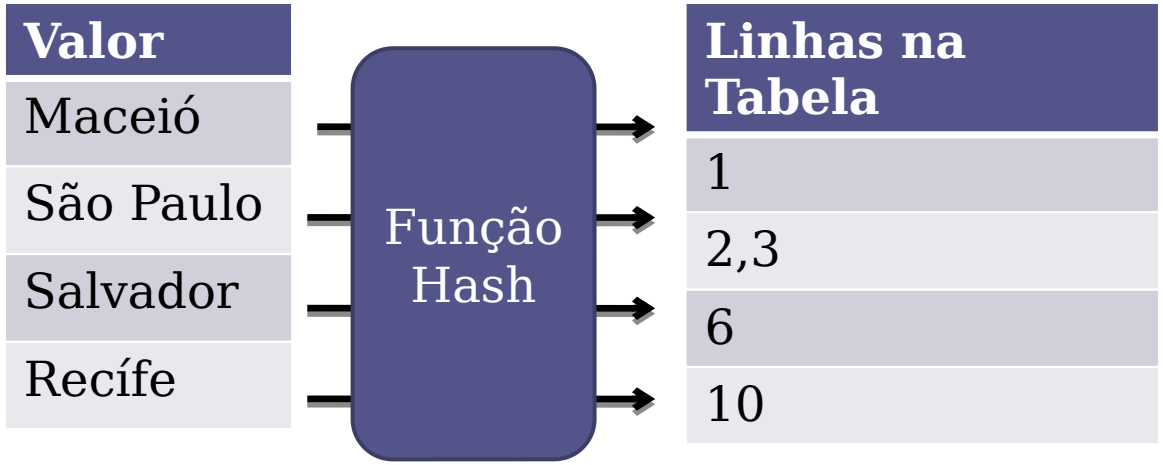
```
CREATE INDEX nome_index ON tabela USING btree(campo_da_tabela);
```

Exemplo:

```
CREATE INDEX pontuacao_index ON jogadores USING btree(pontuacao);
```

Tipos de Índices – Hash

- Hash – funciona bem com buscas de igualdade, onde no select se busca o valor exato.
  - Pode não ser muito usado, pois caso o banco sofre um *crash*, será preciso recriar todos os índices que usam hash.
  - Durante a busca, o valor buscado receberá um calculo de uma função em Hash revelando exatamente a linha onde o campo se encontra:



Exemplo:

SELECT * FROM clientes
WHERE cidade = ‘Recífe’

Tipos de Índices – Hash

- Hash – Aplica é aplicado quando a consulta envolver um valor exato (=)
- Para criar um index de Hash, usamos:

```
CREATE INDEX nome_index ON tabela USING hash(campo);
```

Exemplo:

```
CREATE INDEX cpf_index ON usuarios USING hash(cpf);
```

Tipos de Índices – GiST

- GiST (Generalized Search Tree) – Realiza a construção dos índices através de arvores equilibradas usando operações avançadas, como figuras geométricas, por permitir buscas mais rápidas de registros de vizinhos mais próximos.

- Exemplo de comando aceito:
  - SELECT * FROM lugares ORDER BY localizacao **<->** point '(101,456)' LIMIT 10;

Tipos de Índices – GiST

- GiST (Generalized Search Tree) – Para criar um índice usando o algoritmo GiST, usamos:

```
CREATE INDEX nome_index ON tabela USING gist (campo);
```

Exemplo:

```
CREATE INDEX cep_index ON usuarios USING gist (cep);
```

Tipos de Índices – GIN

- GIN (Generalized Inverted Index) – Tem a capacidade de manipular mais de um valor por registro como um array NO mesmo registro.

- O comando para realizar essa busca é:

```
CREATE INDEX nome_index ON tabela USING gin(campo1, campo2...);
```

Exemplo:

```
CREATE INDEX busca_usuario_index ON usuarios USING gin(nome, cpf);
```

Questões

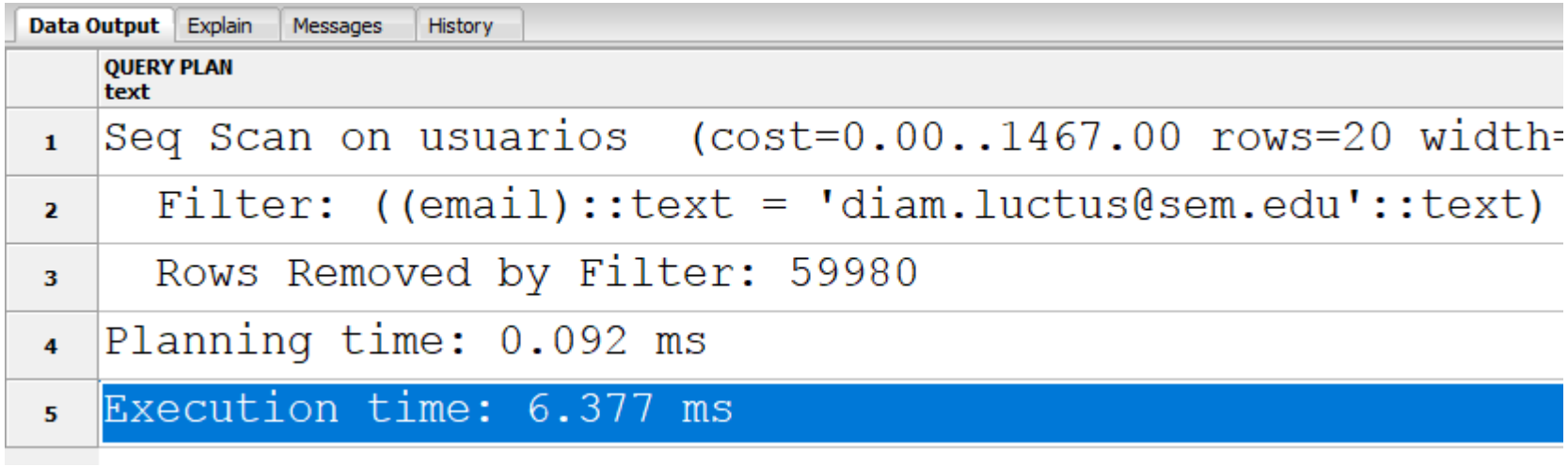
- 1 - Qual é a vantagem de utilizar indexação?
  - Possibilitar uma busca mais rápida, evitando realizar uma busca sequencial em toda tabela;
- 2 – Qual é o comando para criar uma indexação?
  - CREATE INDEX nome_do_index ONnome_tabela (nome_campo)
  - CREATE INDEX nome_do_index ON nome_tabela USING algoritmo(nome_campo)
- 3 – Um sistema usa com frequência uma busca: SELECT * FROM PESSOAS WHERE CPF = 'x', qual algoritmo de indexação melhor se adequa para ser usado para uma busca com valores exato?
  - HASH
- 4 – Qual desses algoritmos não fazem parte do grupo de indexação do Postgres?
  - B-TREE
  - GiST
  - GIN
  - QUEUE
  - HASH

Questões

- Usem o Banco de Dados Fornecido e realizem a seguinte busca:

```
EXPLAIN ANALYSE  SELECT * FROM usuarios
                 WHERE email = 'diam.luctus@sem.edu';
```

- O Comando EXPLAIN junto ao atributo ANALYSE iria fazer uma analise da busca realizada retornando o tempo médio de execução:

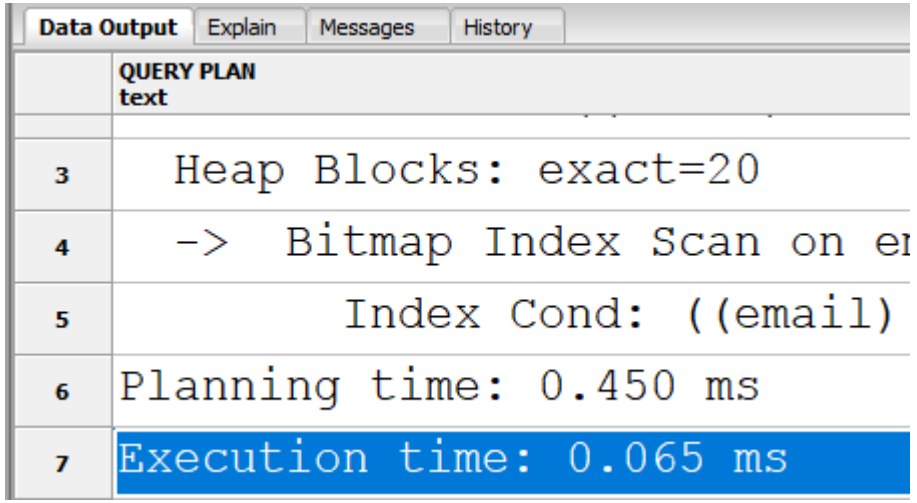


Questões

- Após a checagem, adicione uma index de hash ao campo de email:

```
CREATE INDEX email_index ON usuarios USING
hash(email);
```

- Após isso, realize novamente o comando EXPLAIN ANALYSE no SELECT anterior e veja o novo tempo de resultado:



Questões

- Caso deseje remove rum índice, basta executar o comando:

```
DROP INDEX nome_inde
           x
```

- Se o índice aumenta a velocidade, por que não tornamos todos campos indexados?
  - Pois perderíamos desempenho na hora de inserir, atualizar ou remover dados, uma vez que esse processo precisaria ser feito em diferentes estruturas.